How Close Is Too Close? The Role of Feature Attributions in Discovering Counterfactual Explanations

Anjana Wijekoon(✉), Nirmalie Wiratunga, Ikechukwu Nkisi-Orji, Chamath Palihawadana, David Corsar, and Kyle Martin

School of Computing, Robert Gordon University, Aberdeen, Scotland
a.wijekoon1@rgu.ac.uk

Abstract. Counterfactual explanations describe how an outcome can be changed to a more desirable one. In XAI, counterfactuals are "actionable" explanations that help users to understand how model decisions can be changed by adapting features of an input. A case-based approach to counterfactual discovery harnesses Nearest-unlike Neighbours as the basis to identify the minimal adaptations needed for outcome change. This paper presents the DisCERN algorithm which uses the query, its NUN and substitution-based adaptation operations to create a counterfactual explanation case. DisCERN uses Integrated Gradients (IntG) feature attribution as adaptation knowledge to order substitution operations and to bring about the desired outcome with as few changes as possible. We present our novel approach with IntG where the NUN is used as the baseline against which the feature attributions are calculated. DisCERN also uses feature attributions to identify a NUN closer to the query, and thereby minimise the total change needed, but results suggest that the number of feature changes can increase. Overall, DisCERN outperforms other counterfactual algorithms such as DiCE and NICE in generating valid counterfactuals with fewer adaptations.

Keywords: Counterfactual XAI · Feature attribution · Integrated Gradients · Adaptation

1 Introduction

The use of "similar solutions to solve similar problems" naturally promotes an interpretable reasoning strategy [1]. Exemplar or prototype driven Explainable AI (XAI) methods are able to conveniently use the neighbourhood of similar problems to formulate explanations [2]. For instance a Nearest-like Neighbours (NLNs) based explainer extracts factual information to form an explanation from the similarity between the current problem and it's neighbourhood [3].

This research is funded by the iSee project (https://isee4xai.com) which received funding from EPSRC under the grant number EP/V061755/1.

M. T. Keane and N. Wiratunga (Eds.): ICCBR 2022, LNAI 13405, pp. 33–47, 2022.
https://doi.org/10.1007/978-3-031-14923-8_3

Research has shown that similarity metrics guided by feature selection and weighting [4] can significantly improve the relevance and profiling [5] of NLNs.

Whilst a factual explanation responds to "Why" questions, counterfactuals respond to "Why-Not" type queries. Counterfactuals are regarded as more intuitive for people compared to factual explanations because they present alternatives to reality with more desirable outcomes [6]. The prevailing CBR approach to counterfactual discovery harnesses similarities to Nearest-unlike Neighbours (NUN), i.e. similar cases with different class labels (see Fig. 1) [7,8]. A NUN represents potential changes to the current problem, with feature attribution prioritising the changes that, when actioned, can lead to a different outcome [8,9]. Focusing on a small number of key "actionable" features is more desirable from a practical standpoint, and has the benefit of reducing the recipient's cognitive burden for understanding the counterfactual.

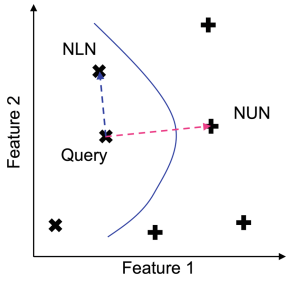


Fig. 1. Nearest-like and nearest-unlike neighbours in 2D space. Explaining the predicted class label of a query based on its similarity to the NLN is a factual explanation, and based on its dissimilarity to the NUN is a counterfactual.

Counterfactual discovery can be viewed as a search in the space of possible feature values guided by feature attribution weights. Here the challenge is to find a counterfactual with minimum feature changes to the query to achieve the needed outcome change. To address this we present DisCERN, a NUN-based counterfactual discovery algorithm that applies substitution-based adaptation operations [10] informed by feature attributions from Integrated Gradients (IntG) [11]. Specifically we seek answers to the following research questions:

- Can the query and NUN case pair form a suitable limit for the integral interval when calculating feature attributions for counterfactual discovery from IntG; and to
- What extent can the approximated integral intervals be used as perturbations to discover counterfactuals closer to the NUN?

The rest of the paper is organised as follows. Section 2 reviews related literature followed by Sect. 3 presenting the DisCERN Algorithm using the IntG feature attribution method for counterfactual discovery. Section 4 presents evaluation methodologies, the datasets and performance metrics with results in Sect. 5. Conclusions and future work appear in Sect. 6.

2 Related Work

Case-based Reasoning (CBR) [7,8] and optimisation techniques [12,13] have been the pillars of discovering counterfactuals. Recent work in CBR has shown how counterfactual case generation can be conveniently supported through the case adaptation stage, where query-retrieval pairs of successful counterfactual explanation experiences are used to create an explanation case-base [7]. Unlike the CBR approach, optimisation techniques like DiCE [12] train a generative model using gradient descent to output multiple diverse counterfactuals. Both approaches uphold two key requirements of good counterfactuals which are: maximising the probability of obtaining the desired label (i.e. different from current label); and minimising the distance (similar to current problem). Additionally the DiCE optimisation also maximises diversity amongst multiple alternative counterfactuals. With the CBR approach additional counterfactuals can be identified by increasing the neighbourhood. In our work presented here, we also adopt a CBR approach to finding counterfactuals, opting for a substitution-based adaptation technique informed by actionable feature recommendations from feature attribution techniques. In doing so, we avoid the need to create similarity-based explanation case-bases, yet maintain the advantage of locality-based explanations which ensure plausible counterfactuals that are often harder to guarantee with optimisation methods.

Feature attribution techniques are used in XAI to convey the extent to which a feature contributes to the predicted class label, with a higher weight indicating higher significance of the feature to the model's decision. Both model-agnostic and model-specific methods are found in literature. Model-agnostic approaches include LIME [14] and SHAP [15]; model-specific approaches often refer to Gradient based techniques, such as DeepLift [16] and Integrated Gradients [11]. Given a query, Integrated Gradients (IntG) are feature attributions calculated as the cumulative change of gradients with respect to interpolated estimates bounded by a baseline and the query. This baseline is formalised as the input where the prediction is neutral. Originally IntG opted to select an all zero input as the baseline for images and text [11], while [17] proposed a mask optimisation approach. Instead in this paper we explore two approaches to selecting the baseline specific to counterfactual discovery based on the uncertainty of the black-box model and the desired outcome change. While the flexibility to use different baselines makes IntG suitable for feature attribution during counterfactual discovery, it limits our approach to explaining only gradient descent optimised models.

3 DisCERN

Consider a neural network classifier F trained to predict the label y for a given input data instance x. The query instance x has m features where the i^{th} feature is denoted by x_i and the label predicted by the classifier for x is $F(x)$. The optimal counterfactual for x is a data instance $\hat{x}$ where $\hat{y}$ is a more desirable label (i.e. $\hat{y} \neq y$) and is closest to x in the feature space. DisCERN's counterfactual discovery for x has the following steps:

1. find the Nearest-unlike Neighbour (NUN), x';
2. compute the feature attribution weight w_i for each x_i of query x, using a feature attribution method Φ. Order the list of features in descending order of their feature attributions, such that the first feature contributes the most to $F(x)$; and
3. iterate over the list of ordered features and at each step, a feature of the query is substituted with the corresponding feature from the NUN, which incrementally forms the adapted query $\hat{x}$. Repeat until $F(\hat{x}) \neq y$, at which point, $\hat{x}$ is selected as the counterfactual for query x.

3.1 Nearest-Unlike Neighbour

DisCERN considers the NUN, x', as the basis for discovering the optimal counterfactual $\hat{x}$ for the query x. Given a query, label pair (x, y) and the training dataset X, a function $\mathcal{N}$ retrieves the query's NUN, (x', y'):

$$\begin{gathered} \mathcal{N} = \underset{(x^i, y^i) \in X}{\arg\min}\, d(x, x^i); y^i \neq y \\ (x', y') \leftarrow \mathcal{N}((x, y), X) \end{gathered} \tag{1}$$

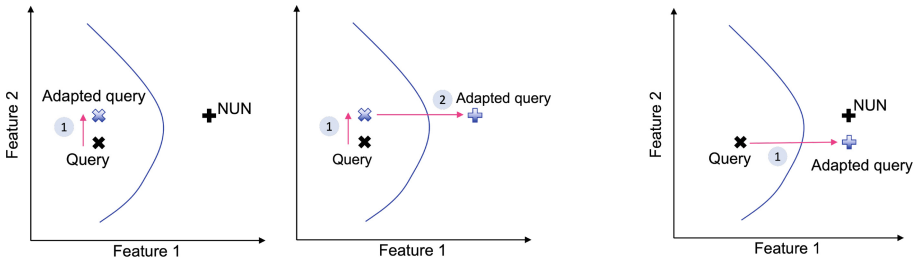
$\mathcal{N}$ calculates the distance $d(.)$ between the query and each candidate (x^i, y^i) in X, and returns the closest (minimum distance) candidate with the desired label, y' (i.e. $y^i \neq y$). Figure 1 illustrates a 2D ($m = 2$) feature space for a binary classed problem. It shows, for a given query, how the NUN (with the desired class label y') appears close to its decision boundary. Although the NUN is a valid counterfactual, there could be an alternative that is closer to the query. Accordingly, DisCERN applies adaptation operations that are bounded by both the query and the NUN to form the counterfactual.

3.2 Feature Ordering by Feature Attribution

Given a query x and its NUN, x', the adaptation step can be as simple as randomly selecting a feature at a time from x' into query $\hat{x}$. Consider the two adaptations presented in Fig. 2. Option 1 (Fig. 2a) needs two substitutions to find its counterfactual; in contrast Option 2 (Fig. 2b) needs just one substitution. This highlights the importance of feature ordering when the query is being iteratively adapted.

Feature attribution techniques identify important features that contributed to a query's class prediction. These attributions can be quantified as weights to enforce an ordering of features. Global feature weighting methods, such as Chi2, and more recent feature attribution explainer methods, such as LIME, SHAP and IntG; are all suitable sources of feature weights. Given feature attributions, w, features are ordered for adaptation in descending order of their feature attributions. The order of the resulting feature indices highlights the features that contribute the most to the current class label. Equation 2 formalises the partial order condition applied to obtain this list of feature indices.

$$x_i \preceq_{\mathcal{R}} x_j \iff \mathcal{R} :: w_i \geq w_j \tag{2}$$



(a) $f_2 \rightarrow f_1$ (b) $f_1 \rightarrow f_2$

Fig. 2. Adaptations based on different feature orderings

3.3 Substitution-Based Adaptation

We illustrate in Fig. 3 the use of feature attributions to adapt query x to its counterfactual $\hat{x}$ following three ordered substitutions. Features of the query are ordered by their attributions where blue and orange colours indicate attributions towards and against the query label. Adaptation involves a series of copy operations from the NUN (from $\mathcal{N}$). Here the adapted query $\hat{x}$ becomes the counterfactual once $\mathrm{F}(\hat{x})$ outputs the same class as the NUN. In the worst case scenario, the counterfactual is equal to the NUN (i.e. $\hat{x} = x'$) where DisCERN performed m number of substitution operations. In an average case, DisCERN performs only n number of substitution operations where $1 \leq n \leq m$.

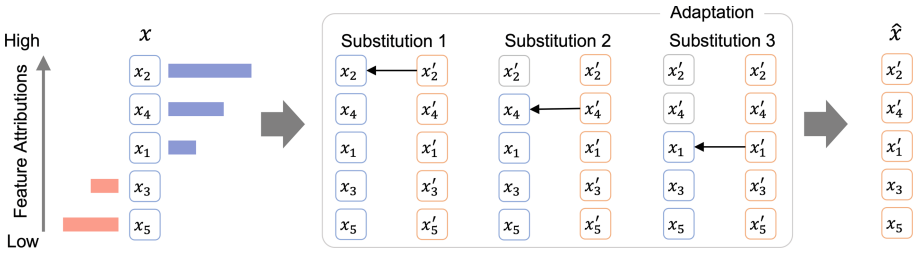


Fig. 3. Substitution-based adaptation operations

3.4 Integrated Gradients for DisCERN

Integrated Gradients (IntG) is a gradient based approach to finding feature attribution weights [11]. An attribution is calculated as the sum of gradients on data points occurring at sufficiently small intervals along the straight-line path from a baseline, x', to the query, x. These intermediate data points are synthetic, perturbed instances between a baseline and the query. With $\mathcal{A}$, number of perturbations, the α^{th} perturbation, x^{α}, can be calculated as in Eq. 3.

$$x^{\alpha} = x' + \frac{\alpha}{\mathcal{A}} \times (x - x') \tag{3}$$

Here α controls the number of integral interval steps the baseline has taken towards the query x. The amount of perturbation is controlled by both the integral step $\frac{\alpha}{\mathcal{A}}$ and the difference between query and baseline. Accordingly a perturbed data instance can be viewed as the baseline taking α number of integral interval steps towards the query x. For symbolic features in tabular data, this requires converting symbolic values into nominal values before applying the integration interval step in Eq. 3. Figure 4 presents an example in the 2D space,

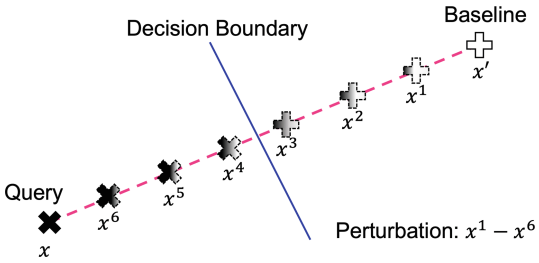


Fig. 4. Perturbation created between the baseline and the query

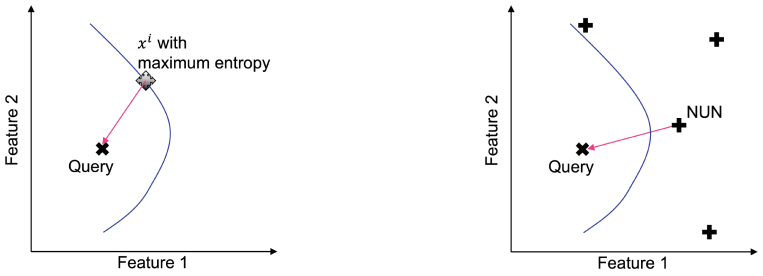
where six perturbations are created on the straight-line path between a query and a baseline. Given the set of perturbations, an attribution weight w_i for feature x_i, is calculated as in Eq. 4.

$$w_i = (x_i - x'_i) \times \sum_{\alpha=1}^{\mathcal{A}} \frac{\partial x^{\alpha}}{\partial x_i} \tag{4}$$

In practice, a large number of perturbations is preferred because the summation of gradients is a discrete approximation of a continuous integration as discussed in [11].

An important decision for IntG feature attribution is the choice of the baseline. Typically an area of uncertainty provides a suitable place to form this baseline. With respect to a neural network classifier a data instance for which the classifier predicts similar probabilities to all class labels at the Softmax activation layer would be an ideal baseline choice. More formally, given the classifier F and a data instance x^i in X, the uncertainty of F in predicting the label of x^i is measured by an entropy $H(x^i|F)$ score. Therefore the chosen baseline, x', is the x^i which maximises entropy from all data instances.

$$x' \leftarrow \underset{x^i \in X}{\arg\max}\ H(x^i|F); x^i \in X \tag{5}$$



(a) Baseline that maximises entropy for vanilla feature attribution

(b) Baseline that minimises distance for guiding counterfactual discovery

Fig. 5. Choice of baselines for IntG

Visually, the selected baseline, x', is either on the decision boundary or very close to it (Fig. 5a). Therefore the resultant feature attributions from a maximal entropy informed baseline, will identify feature values that caused the query to move away from the decision boundary where uncertainty is highest to its current class label.

In DisCERN, we want to choose a baseline for the straight line approximations to better suit our task of counterfactual discovery where feature attributions should highlight actionable features. Additionally this baseline data instance should be closer to the query in the feature space to minimise the changes needed to cross the decision boundary for class change. This is different from having a constant baseline that on average is an uncertain instance to all data points. Instead with DisCERN the baseline should change according to the locality of the query. Therefore the natural baseline choice for DisCERN is to have the NUN as the baseline. More formally, given the NUN function $\mathcal{N}$, and training set X, the baseline x' is retrieved as follows:

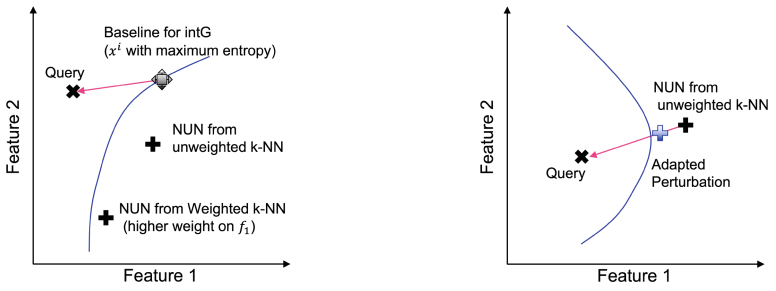
$$x' \leftarrow \mathcal{N}((x, y), X) \tag{6}$$

Figure 5b illustrates an example where a NUN is selected as the baseline. Here too the baseline by being close to the query is also in an area of uncertainty. However instead of a globally uncertain area, the NUN ensures that this uncertainty is locally relevant to the query to action the desired class change. Accordingly, the feature attributions can be interpreted as the feature contributions that action the query change from one side of the decision boundary to the other.

3.5 Bringing the NUN Closer

DisCERN's adaptations are bounded by the NUN such that in the worst case scenario, the adaptation-based counterfactual discovery results in the NUN itself (i.e. when all NUN values are copied over to the query). Typically, DisCERN would discover a counterfactual after performing only a subset of the total substitutions suggested by the NUN. This means that an adapted query, with just

a small subset of feature value substitutions from the NUN, can be pushed over the boundary to the desired class label. Therefore it is interesting to consider whether an "even closer" NUN can be identified by focusing on just a subset of important features. Here we study two methods to bring the NUN closer to the boundary using: a weighted k-NN retrieval ($\mathcal{N}_w$); and from adapted perturbations ($\mathcal{N}_p$). The NUN found by the former is a true data instance whilst the latter is a perturbed "synthetic" data instance. We will refer to the unweighted k-NN retrieval of the NUN discussed in Sect. 3.1 as $\mathcal{N}$ in order to contrast that with the retrieval methods used by the weighted and perturbed methods aimed at recognising a NUN that is closer to the query.



(a) Weighted k-NN retrieval to find NUN (b) Adapted IntG Perturbation

Fig. 6. Bringing NUN closer

Weighted K-NN Retrieval uses a feature attribution technique to select the subset of features to attend to when calculating distance. It then returns the NUN, x', for query, x, as follows:

$$\begin{gathered} w = \phi(x, ...) \\ \mathcal{N}_w = \underset{(x^i, y^i) \in X}{\arg\min}\, d(x, w, x^i); y^i \neq y \\ (x', y') \leftarrow \mathcal{N}_w((x, y), \phi, X) \end{gathered} \tag{7}$$

Here ϕ returns the feature attributions w by adopting a method such as LIME [14], SHAP [15] or IntG [11]. The ... denotes any other requirements of the ϕ to generate feature attributions (such as method specific hyper-parameters). Note that when ϕ is IntG, a baseline must be specified before feature attribution weights can be computed. This weighted retrieval is illustrated in Fig. 6a for a given query x. The baseline for IntG here is an instance that maximises uncertainty, as measured by entropy in Eq. 5 (vanilla version). In our 2D space the feature attributions for Features 1 and 2 are $w_1 \geq w_2$, as such $\mathcal{N}_w$ will return the new NUN which is closer to the query with respect to f_1.

Adapted Perturbation approach creates a synthetic data instance by creating a perturbed instance between the NUN (retrieved by $\mathcal{N}$) and the query, x, which is closer to the query. This is illustrated in Fig. 6b with an example in 2D space where the "blue" coloured cross shows an ideal synthetic perturbed NUN that is closer to the query compared to the actual NUN. The discovery of this perturbed NUN, x', can be formalised as follows:

$$\begin{gathered} \alpha^* = \arg\max_{\alpha \in \mathcal{A}} \alpha; F(x^\alpha) \neq y \\ \mathcal{N}_p = \begin{cases} x'_i & \text{if } i \text{ is a symbolic feature} \\ x_i^{\alpha*} & \text{otherwise} \end{cases} \\ (x', y') \leftarrow \mathcal{N}_p((x, y), \mathcal{N}, X) \end{gathered} \tag{8}$$

First, an $\mathcal{A}$ number of perturbations are created using Eq. 3 from the baseline to the query (see Fig. 4). The black box model, F, is used to predict the class for each perturbed data instance in turn. The selected perturbation, α^*, is the closest to the boundary with a prediction matching the desired class. Depending on the feature type (i.e. numeric or symbolic), values from the baseline, x', and $x^{\alpha*}$ are used to form the adapted perturbation. Specifically, symbolic features are substituted from x' and numerical features from the selected perturbation $x^{\alpha*}$. Note that the latter will have interpolated numeric values discovered as part of the perturbations. As a result, $\mathcal{N}_p$ returns a synthetic NUN, which preserves plausibility [6] when used in DisCERN's adaptation operations.

In Fig. 6b the adapted perturbation is on the straight-line path between the NUN from $\mathcal{N}$ and the query. Here both Features 1 and 2 are numeric. However with categorical features, the resulting adapted perturbation may not fall on the straight-line, yet we confirm that $F(x^{\alpha*}) \neq y$ (i.e. ensures a valid NUN from a different class to that of the query).

4 Evaluation

Three questions answered in the evaluations as follows:

- how does IntG compare to other feature attribution methods for counterfactual discovery with DisCERN?;
- how does DisCERN with IntG feature attributions compare against other counterfactual discovery algorithms in literature?; and
- what is the impact of bringing NUN closer in NUN-based counterfactual discovery?

4.1 Datasets

Five public domain datasets were used - Loan and Income datasets are from Kaggle; and Credit, Cancer and ICU datasets are from the UCI Machine Learning Data Repository. Table 2 summarises the properties of each dataset and

Table 1. Dataset details

Datasets	Dataset description	Explanation need	Labels
Loan	Predicts if a loan application is approved or not	What changes to the application will help to get the loan approved?	Q: Rejected CF: Approved
Income	Predicts if a persons income is higher or lower than 50K based on Census data	What changes will help an adult get an income over 50K?	Q: $\leq$50K CF: $\geq$50K
Credit	Predict if a credit card application is successful or not	What changes to the application will help to get a credit card?	Q: Rejected CF: Approved
Cancer	Predict the level of lung cancer risk of a patient based on demographic and triage data	How to reduce lung cancer risk of a patient?	Q: High risk or Medium risk CF: Low risk
ICU	Predict patient outcome in the Intensive Care Unit (ICU) based on demographic and diagnostic data	How to prevent the death of a patient in the ICU?	Q: Deceased CF: Discharged

the explanation needs as questions. Our evaluations consider all features actionable due to the lack of contextual data available. The labels column shows two class labels: the desirable (CF); and less desirable (Q). Typically a counterfactual explanation is needed to explain how a desirable situation could have been achieved when the black box model prediction is the less desired class.

Table 2. Dataset properties

Datasets	Loan	Income	Credit	Cancer	ICU
Features	69	14	14	12	107
Categorical features	8	8	8	11	8
Classes	2	2	2	3	2
Data instances	342865	45222	653	427	6238
Negative test instances	22543	11216	121	109	1003
RF test accuracy (%)	99.51	85.66	77.31	87.94	81.70
NN test accuracy (%)	99.66	85.02	75.00	87.23	73.08

4.2 Experiment Setup

The experimental pipeline to measure performance of a counterfactual discovery method appears in Fig. 7. First the dataset is split as 2/3 train and 1/3 test data. Next, a black-box classifier is trained and used to populate a case-base of labelled train data. In this paper we consider two classifiers: Random Forest

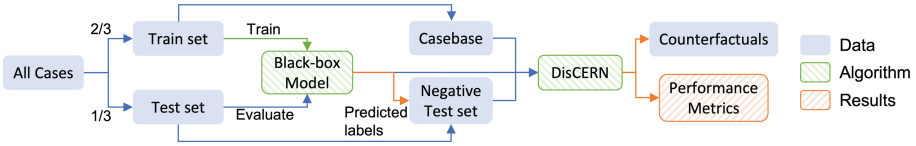


Fig. 7. Experiment setup

and Neural Networks. Test set performance of these 2 classifiers are in Table 2. The test instances classified into the less desirable class are the queries which form the Negative test set (see Table 1 for more details). Finally, the counterfactual discovery algorithm uses the classifier and the populated case-base to find counterfactuals for all query instances in the Negative test set. DisCERN uses Euclidean distance as $d(.)$; and it is available publicly on GitHub[1].

4.3 Performance Measures for Counterfactual Explanations

Validity (V) measures the percentage for which an algorithm is able to find a counterfactual with the desired label [12]. If N is the total number of queries and N_v the number of counterfactuals found, then $V = \frac{N_v}{N} \times 100$.

Sparsity ($\#F$) is the average number of feature differences between a query (x) and its counterfactual ($\hat{x}$) explanation [12]. Here m is the number of features.

$$\#F = \frac{1}{N_v \times m} \sum_{j=1}^{N_v} \sum_{i=1}^{m} 1_{[\hat{x}_i \neq x_i]} \tag{9}$$

Proximity ($\$F$) measures the mean feature-wise distance between a query and its counterfactual explanation [12]. The sum of normalised feature differences are averaged over $\#F$ and the number of counterfactuals (N_v).

$$\$F = \frac{1}{N_v \times \#F} \sum_{j=1}^{N_v} \sum_{i=1}^{m} (|\hat{x}_i - x_i|) \tag{10}$$

Datasets with more categorical features will have higher $\$F$ (difference is always 1), meaning, $\$F$ is not comparable across datasets. Note that higher values of V and smaller values of $\#F$ and $\$F$ are preferred.

5 Results

5.1 A Comparison of Feature Attribution Techniques

Feature attributions determine the order in which values are copied from NUN to query. This adaptation knowledge guides the value substitution operations until

[1] https://github.com/RGU-Computing/DisCERN-XAI.

a valid counterfactual is discovered. In Fig. 8 we compare DisCERN with five feature attribution variants for ordering substitution operations: **Random** (i.e. no adaptation knowledge); $\mathcal{X}^2$ global feature selection; **LIME** local surrogate relevance weights, **SHAP**ely values from the KernelSHAP implementation; and **IntG** with the NUN ($\mathcal{N}$) baseline. Here, DisCERN uses the NUN from unweighted k-NN retrieval (i.e. $\mathcal{N}$) In Fig. 8, 5 marker shapes corresponds to each of the 5 algorithms, and 5 marker colours relate to results corresponding to each dataset. When sparsity and proximity are low the markers appear closer to the origin - this is desirable. Note that for visual clarity the x-axis is in log scale.

IntG achieves the lowest sparsity while random and $\mathcal{X}^2$ techniques achieves lower proximity. IntG's use of the NUN baseline has successfully contributed to reducing sparsity, i.e. IntG's feature attributions had significantly reduced the number of adaptation operations, when compared to using feature attributions from LIME and SHAP. Generally we found that with all datasets, making larger changes to feature values ($\$F$) leads to discovering a counterfactual with fewer feature substitutions ($\#F$). Therefore when we consider the total change ($\#F \times \$F$), IntG is best on all datasets except on Loan, where $\mathcal{X}^2$ was better. We also see that IntG had higher proximity (0.72 compared to the rest in a range of 0.28–0.39) on the Cancer dataset, but it had significantly lower sparsity (fewer features adapted), with almost half that of the other attribution methods (log scale can mask this difference). This can be a result of IntG selecting categorical features compared to other techniques.

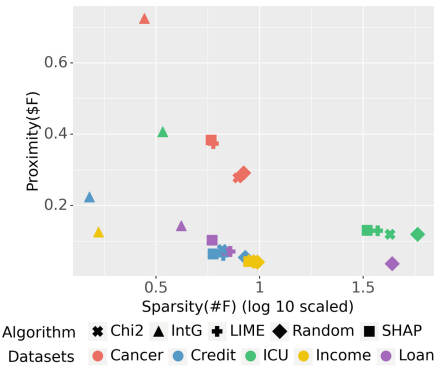


Fig. 8. Feature attribution results

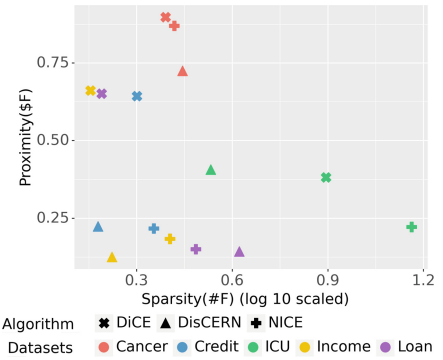


Fig. 9. Counterfactual algorithm results

5.2 A Comparison of Counterfactual Discovery Algorithms

A sparsity and proximity comparison of DisCERN (using $\Phi = IntG$) with popular counterfactual discovery algorithms from literature appear in Fig. 9. DiCE is a generative algorithm that discovers counterfactuals by optimising a randomly initialised input to maximise diversity and minimise sparsity and proximity [12].

Table 3. Validity and total change between counterfactual discovery algorithms

Algorithm	Validity					Total Change				
	Loan	Income	Credit	Cancer	ICU	Loan	Income	Credit	Cancer	ICU
DiCE	99.20	92.55	80.99	**100**	33.80	1.009	0.945	1.286	2.207	2.991
NiCE	97.70	92.50	85.12	82.57	82.35	**0.461**	0.466	0.491	2.278	3.230
DisCERN	**100**	**100**	**100**	**100**	**100**	0.599	**0.209**	**0.338**	**2.014**	**1.387**

NICE is a NUN-based counterfactual discovery algorithm that uses a reward function to minimise sparsity, proximity and to preserve plausibility [8].

Overall minimum proximity is seen with DisCERN (lowest on credit, income and loan). DiCE is better for sparsity (lowest on cancer, income and loan) but at the expense of proximity, which requires larger value adaptations (i.e. higher proximity) of the query applied to a fewer number of features (i.e. lower sparsity). Between NUN-based algorithms NICE and DisCERN, the lower sparsity and proximity achieved with DisCERN makes it a better candidate for counterfactual discovery.

Crucial to counterfactual explainers is the ability to find a valid counterfactual situation (i.e. one with the desired class). Results in Table 3 show that DisCERN dominates in this respect, whereby the NUN guided counterfactual is 100% valid. This is in comparison to DiCE, which uses a generative model given a randomly initialised input and NICE, which like DisCERN, uses a NUN as the basis but uses a reward function to find a counterfactuals. As a result, DiCE and NICE are likely to either fail to generate a counterfactual, or fail to generate a counterfactual that does belong to the desired class (in a multi-class setting). Note that for DiCE and NICE results in Fig. 9, we have reported averaged results using only those queries for which a valid counterfactual was discovered (N_v). We observed a trade-off between sparsity and proximity (in Fig. 9), and accordingly these are combined into total change ($\#F \times \$F$) in Table 3. Here DisCERN is best on all datasets (except Loan where it is marginally worse) as it manages to better balance proximity and sparsity requirements.

5.3 Impact of Bringing NUN Closer

Figure 10 compares DisCERN counterfactual discovery with the weighted ($\mathcal{N}_w$) and non weighted kNN ($\mathcal{N}$) NUN retrieval methods. The additional adapted perturbation ($\mathcal{N}_p$) results is included for IntG feature attribution method. Here in each evaluation DisCERN ensures $\Phi = \phi$. For example, if DisCERN feature ordering (Φ) is using SHAP feature attributions, weighted k-NN retrieval also uses SHAP weights ($\phi = SHAP$) to find the closer NUN. We note that $\mathcal{N}_w$ approach was not evaluated with the Cancer dataset since we were unable to find a valid baseline that maximises entropy in the comparably smaller train set.

Overall $\mathcal{N}_p$ with IntG and $\mathcal{N}_w$ with SHAP and IntG show increased sparsity and decreased proximity compared to $\mathcal{N}$. Lower proximity is a result of bringing the NUN closer with respect to a subset of important features. Here the resulting

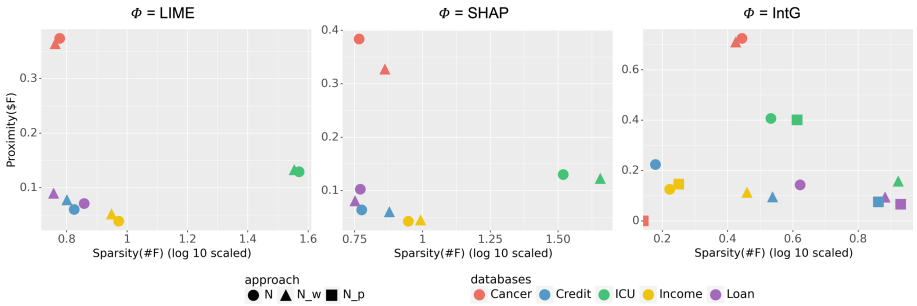


Fig. 10. Results on alternative approaches to form closer NUNs

NUN will have smaller value differences (decreased proximity) with the query. Therefore, initial substitutions will only make smaller actionable changes to the query. These initial features were already part of the subset that influenced the weighted kNN retrieval. As a result, the query will need an increased number of feature changes (increased sparsity). However, empirical evidence remains inconclusive for $\mathcal{N}_w$ with LIME. This might be explained by LIME being a local surrogate attribution method. Accordingly, $\mathcal{N}_w$ with LIME returns a NUN closer to the decision boundary of the local surrogate model, not of the global model. As a result, DisCERN makes larger changes to the query (i.e. increased proximity) in earlier substitutions which decreases the number of feature changes needed (decreased sparsity). In contrast, SHAP or IntG feature attributions seems globally faithful and better aligned to the black box decision boundary. Similarly, adapted perturbation based NUN is also globally faithful which results in increased sparsity and decreased proximity.

6 Conclusions

The XAI DisCERN algorithm discovers counterfactual explanations by applying a set of substitution-based adaptation operations to the NUN. These adaptations help create counterfactual explanations for black box predictions. DisCERN uses feature attribution weights, firstly to guide adaptation choices and, secondly to identify NUNs closer to the decision boundary. Both lead to valid counterfactuals from minimally adapted queries. The use of integrated gradients for feature attribution by limiting the integral approximation to the query and NUN is a unique contribution of this paper. Results suggests that this approach is more effective in guiding counterfactual discovery compared to feature attributions that do not consider both the desired and current prediction boundary locality.

An interesting study of where to position the NUN in support of counterfactual discovery found that it was possible to significantly minimise the total actionable changes required to create the counterfactual explanation, but also increased the number of feature changes needed for the desirable class change. In conclusion, DisCERN discovers effective counterfactuals, and provides the flexibility to select NUN and feature attribution technique based on the AI model

and the preference of feature changes in the problem domain. Future work will look to expand DisCERN to generate counterfactual explanations that can also satisfy immutable and constrained feature requirements.

References

1. Li, O., Liu, H., Chen, C., Rudin, C.: Deep learning for case-based reasoning through prototypes: a neural network that explains its predictions. In: 32nd AAAI Conference on Artificial Intelligence, pp. 3530–3537 (2018)
2. Arrieta, A.B., et al.: Explainable artificial intelligence (XAI): concepts, taxonomies, opportunities and challenges toward responsible AI. Inf. Fusion **58**, 82–115 (2020)
3. Kenny, E.M., Keane, M.T.: Twin-systems to explain artificial neural networks using case-based reasoning: comparative tests of feature-weighting methods in ANN-CBR twins for XAI. In: IJCAI-19, pp. 2708–2715, IJCAI (2019)
4. Wettschereck, D., Aha, D.W., Mohri, T.: A review and empirical evaluation of feature weighting methods for a class of lazy learning algorithms. Artif. Intell. Rev. **11**(1), 273–314 (1997)
5. Craw, S., Massie, S., Wiratunga, N.: Informed case base maintenance: a complexity profiling approach. In: AAAI, pp. 1618–1621 (2007)
6. Byrne, R.M.: Counterfactuals in explainable artificial intelligence (XAI): evidence from human reasoning. In: IJCAI, pp. 6276–6282 (2019)
7. Keane, M.T., Smyth, B.: Good counterfactuals and where to find them: a case-based technique for generating counterfactuals for explainable AI (XAI). In: Watson, I., Weber, R. (eds.) ICCBR 2020. LNCS (LNAI), vol. 12311, pp. 163–178. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58342-2_11
8. Brughmans, D., Martens, D.: Nice: an algorithm for nearest instance counterfactual explanations. arXiv preprint arXiv:2104.07411 (2021)
9. Wiratunga, N., Wijekoon, A., Nkisi-Orji, I., Martin, K., Palihawadana, C., Corsar, D.: Discern: discovering counterfactual explanations using relevance features from neighbourhoods. In: 33rd ICTAI, pp. 1466–1473. IEEE (2021)
10. Craw, S., Wiratunga, N., Rowe, R.C.: Learning adaptation knowledge to improve CBR. Artif. Intell. **170**(16–17), 1175–1192 (2006)
11. Sundararajan, M., Taly, A., Yan, Q.: Axiomatic attribution for deep networks. In: International Conference on Machine Learning, pp. 3319–3328. PMLR (2017)
12. Mothilal, R.K., Sharma, A., Tan, C.: Explaining machine learning classifiers through diverse counterfactual explanations. In: Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency, pp. 607–617 (2020)
13. Karimi, A.-H., Barthe, G., Balle, B., Valera, I.: Model-agnostic counterfactual explanations for consequential decisions. In: International Conference on Artificial Intelligence and Statistics, pp. 895–905. PMLR (2020)
14. Ribeiro, M.T., Singh, S., Guestrin, C.: Why should i trust you? explaining the predictions of any classifier. In: 22nd ACM SIGKDD, pp. 1135–1144 (2016)
15. Lundberg, S.M., Lee, S.-I.: A unified approach to interpreting model predictions. Adv. Neural. Inf. Process. Syst. **30**, 4765–4774 (2017)
16. Li, J., Zhang, C., Zhou, J.T., Fu, H., Xia, S., Hu, Q.: Deep-lift: deep label-specific feature learning for image annotation. IEEE Trans. Cybern. (2021)
17. Qi, Z., Khorram, S., Li, F.: Visualizing deep networks by optimizing with integrated gradients. In: CVPR Workshops, vol. 2 (2019)